Programmable Networks with Synthesis

Ahmed El-Hassany  Petar Tsankov  Laurent Vanbever  Martin Vechev

http://synet.ethz.ch
Network Misconfigurations are Common

Amazon server outage affects millions of companies and causes online chaos

AMAZON’S giant servers crashed today causing chaos for millions of companies and people that use the cloud service and affected everything larger web sites to people’s smart homes and even library catalogues.

Amazon’s massive AWS outage was caused by human error

One incorrect command and the whole Internet suffers.

CloudFlare apologizes for Telia screwing you over

Unhappy about massive outage

The summer of network misconfigurations


21 Jun 2016 at 20:34, Kieren Mc

Networks switch config blunder blamed for US-wide VoIP blackout

Network dropped calls because it was told to

5 Oct 2016 at 11:33, Shaun Nichols
What Makes Network Configuration Hard?

- High-level, global routing requirements
- Low-level, local router configurations
- Multiple interacting routing protocols (OSPF, BGP, ..)
R1: Packets from **N1** to **N2** must follow the path **A → D**

R2: Packets from **N1** to **N3** must cross intrusion detection system (IDS)
Example

R1: Packets from **N1** to **N2** must follow the path **A → D**

R2: Packets from **N1** to **N3** must cross intrusion detection system (IDS)
**Example**

**Network N1**

- **R1:** Packets from **N1** to **N2** must follow the path **A → D**

**Network N3**

- **R2:** Packets from **N1** to **N3** must cross intrusion detection system (IDS)

**Network N2**

**Static routes table**

<table>
<thead>
<tr>
<th>Network</th>
<th>NextHop</th>
</tr>
</thead>
<tbody>
<tr>
<td>N3</td>
<td>B</td>
</tr>
</tbody>
</table>

**Add a static route to A configuration**

**Configure A to prefer static routes over OSPF**

**B non-deterministically forwards packets for N3 to either A or C**
**Example**

**Static routes table**

<table>
<thead>
<tr>
<th>Network</th>
<th>NextHop</th>
</tr>
</thead>
<tbody>
<tr>
<td>N3</td>
<td>B</td>
</tr>
</tbody>
</table>

**Router configurations must be such that:**

1. **A** prefers *static routes* over **OSPF**
2. **A** has a static route to **B** for **N3**
3. **A → D → N2** must be lowest cost from **A** to **N2**
4. **B → C → D → N3** must be lowest cost from **B** to **N3**

**R1:** Packets from **N1** to **N2** must follow the path **A → D**

**R2:** Packets from **N1** to **N3** must cross intrusion detection system (IDS)

**IDS**

Lower cost to 5
Current Practice

Problems and Challenges

- **Diversity** in protocol expressiveness
- Protocol **dependencies**
- No **correctness** guarantees

Operators manually configure each router. All routers are configured.
State-of-the-art

**Verification**

*ARC SIGCOMM’16, Batfish NSDI’15, rcc NSDI’05, ...*
State-of-the-art

Verification
ARC SIGCOMM’16, Batfish NSDI’15, rcc NSDI’05, ...

Template-based
RPSL RFC 2622, NETCONF, ...
State-of-the-art

**Verification**
*ARC SIGCOMM’16, Batfish NSDI’15, rcc NSDI’05, ...*

**Template-based**
*RPSL RFC 2622, NETCONF, ...*

**Synthesizing configurations**
*Methane PLDI’17 (BGP only), Propane SIGCOMM’16 (BGP only), Fibbing SIGCOMM’15 (link-state only), ...*
Current Practice

Programmable Networks with Synthesis

How is the behavior of routers captured?

Operators manually configure each router

How to express relevant requirements?

Routing requirements

How to find a configuration that conforms to the requirements?

All routers are configured

Network topology
Capturing Network Behavior

**Key idea:** Express routing protocols, along with their dependencies, in *stratified Datalog*
Datalog (1/2 Graph Reachability)

**Input**

\(\text{link}(N1,A)\)
\(\text{link}(A,B)\)
...

**Program**

\(\text{path}(X,Y) \leftarrow \text{link}(X,Y)\)
\(\text{path}(X,Y) \leftarrow \text{link}(X,Z), \text{path}(Z,Y)\)

**Query**

\(\text{path}(N1,N2)\)?

Can we capture the network’s forwarding plane?
Datalog (2/2 Shortest-path Routing)

**Input**

\[ \text{link}(N1, A, 10) \]

...  

**Program**

\[ \text{path}(\text{Router}, \text{Net}, \text{Net}, \text{Cost}) \leftarrow \text{link}(\text{Router}, \text{Net}, \text{Cost}) \]

\[ \text{path}(\text{Router}, \text{Net}, \text{NextHop}, C_1 + C_2) \leftarrow \text{link}(\text{Router}, \text{NextHop}, C_1), \text{path}(\text{NextHop}, \text{Net}, X, C_2) \]

\[ \text{sp}(\text{Router}, \text{Net}, \text{NextHop}, \text{min}(C)) \leftarrow \text{path}(\text{Router}, \text{Net}, \text{NextHop}, C) \]

\[ \text{fwd}(\text{Router}, \text{Net}, \text{NextHop}) \leftarrow \text{sp}(\text{Router}, \text{Net}, \text{NextHop}, C) \]

**Query**

\[ \text{fwd}(A, N2, ?) \]
Datalog Stratification

A Datalog program $P$ is **stratified** if its rules can be partitioned into sets $P_1,\ldots,P_n$ called strata, such that:

1. for every predicate $p$, all rules with $p$ in their heads are in one stratum $P_i$
2. if a predicate symbol $p$ occurs in a positive literal in $P_i$, then all rules with $p$ in their heads are in a stratum $P_j$ with $j \leq i$
3. if a predicate symbol $p$ occurs in a negative literal in $P_i$, then all rules with $p$ in their heads are in a stratum $P_j$ with $j < i$
Datalog Stratification

**Input**

\[\text{link}(N1, A, 10)\]

...  

**Program**

\[\text{path}(\text{Router, Net, Net, Cost}) \leftarrow \text{link}(\text{Router, Net, Cost})\]

\[\text{path}(\text{Router, Net, NextHop, } C_1 + C_2) \leftarrow \text{link}(\text{Router, NextHop, } C_1), \text{path}(\text{NextHop, Net, } X, C_2)\]

\[\text{sp}(\text{Router, Net, NextHop, min}(C)) \leftarrow \text{path}(\text{Router, Net, NextHop, } C)\]

\[\text{fwd}(\text{Router, Net, NextHop}) \leftarrow \text{sp}(\text{Router, Net, NextHop, } C)\]

**Query**

\[\text{fwd}(A, N2, ?)\]
Routing Requirements

Paths

Packets for traffic class TC must follow the path $r_1 \rightarrow \cdots \rightarrow r_n$

\[ \text{fwd}(r_1, tc, r_2) \land \cdots \land \text{fwd}(r_{n-1}, tc, r_n) \]
Routing Requirements

Paths
Packets for traffic class TC must follow the path $r_1 \rightarrow \cdots \rightarrow r_n$

Traffic isolation
The paths for two distinct traffic classes $tc_1$ and $tc_2$ do not share links in the same direction

$\forall R_1, R_2. \text{fwd}(R_1, tc_1, R_2) \Rightarrow \neg \text{fwd}(R_1, tc_2, R_2)$
Routing Requirements

Paths
Packets for traffic class $TC$ must follow the path $r_1 \rightarrow \cdots \rightarrow r_n$

Traffic isolation
The paths for two distinct traffic classes $tc_1$ and $tc_2$ do not share links in the same direction

Reachability
Packets for traffic class $tc$ can reach router $r_2$ from router $r_1$
Routing Requirements

Paths
Packets for traffic class $TC$ must follow the path $r_1 \rightarrow \cdots \rightarrow r_n$

Traffic isolation
The paths for two distinct traffic classes $tc_1$ and $tc_2$ do not share links in the same direction

Reachability
Packets for traffic class $tc$ can reach router $r_2$ from router $r_1$

Loop-freeness
The forwarding plane has no loops
Network-wide Configuration Synthesis
Network-wide Configuration Synthesis

Network specification $N$
- (OSPF, BGP, MPLS, ...)

Routing requirements $R$
- (isolation, reachability, reliability)

Synthesis problem:
Find a configuration $C$ such that $N$ configured with $C$ satisfies $R$.

Network-wide configuration $C$
- (protocol configurations for routers)

Datalog program $P$

Datalog query $Q$

Problems:
- No input synthesis tools for Datalog
- The general problem is undecidable

Datalog input $I$
Input Synthesis for Datalog

Key idea: Reduce to solving logical constraints
Stratified Datalog Input Synthesis

<table>
<thead>
<tr>
<th>Rule</th>
<th>Datalog Input Synthesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>( link(N1, A, 10) )</td>
<td>( \text{link}(N1, A, ?) )</td>
</tr>
<tr>
<td>( \ldots )</td>
<td>( \ldots )</td>
</tr>
<tr>
<td>( \text{path}(\text{Router}, \text{Net}, \text{Net}, \text{Cost}) \leftarrow \text{link}(\text{Router}, \text{Net}, \text{Cost}) )</td>
<td>( \text{path}(\text{Router}, \text{Net}, \text{Net}, \text{Cost}) \leftarrow \text{link}(\text{Router}, \text{Net}, \text{Cost}) )</td>
</tr>
<tr>
<td>( \text{path}(\text{Router}, \text{Net}, \text{NextHop}, C_1 + C_2) \leftarrow \text{link}(\text{Router}, \text{NextHop}, C_1), \text{path}(\text{NextHop}, \text{Net}, X, C_2) )</td>
<td>( \text{path}(\text{Router}, \text{Net}, \text{NextHop}, C_1 + C_2) \leftarrow \text{link}(\text{Router}, \text{NextHop}, C_1), \text{path}(\text{NextHop}, \text{Net}, X, C_2) )</td>
</tr>
<tr>
<td>( \text{sp}(\text{Router}, \text{Net}, \text{NextHop}, \text{min}(C)) \leftarrow \text{path}(\text{Router}, \text{Net}, \text{NextHop}, C) )</td>
<td>( \text{sp}(\text{Router}, \text{Net}, \text{NextHop}, \text{min}(C)) \leftarrow \text{path}(\text{Router}, \text{Net}, \text{NextHop}, C) )</td>
</tr>
<tr>
<td>( \text{fwd}(\text{Router}, \text{Net}, \text{NextHop}) \leftarrow \text{sp}(\text{Router}, \text{Net}, \text{NextHop}, C) )</td>
<td>( \text{fwd}(\text{Router}, \text{Net}, \text{NextHop}) \leftarrow \text{sp}(\text{Router}, \text{Net}, \text{NextHop}, C) )</td>
</tr>
<tr>
<td>( \text{fwd}(A, N_2, ?) )</td>
<td>( \text{fwd}(A, N_2, D) \wedge \text{fwd}(A, N_3, B) \wedge \ldots )</td>
</tr>
</tbody>
</table>
Input Synthesis for Datalog (via Constraint Solving)

Datalog program $P$

- $\text{path}(X,Y) \leftarrow \text{link}(X,Y)$
- $\text{path}(X,Y) \leftarrow \text{link}(X,Z), \text{path}(Z,Y)$

Datalog query $Q$

- $\text{path}(a,c) \land \neg \text{link}(a,c)$

Generate constraints

- $\forall X,Y. \text{path}_1(X,Y) \iff \text{link}(X,Y)$
- $\forall X,Y. \text{path}_2(X,Y) \iff \left( \text{link}(X,Y) \lor \exists Z. (\text{link}(X,Z) \land \text{path}_1(Z,Y)) \right)$
- $\text{path}_2(a,c) \land \neg \text{link}(a,c)$

Constraints $\Psi$

Constraint solving

- $\text{link}(a,b), \text{link}(b,c), \text{path}_1(a,b), \text{path}_1(b,c), \text{path}_2(a,b), \text{path}_2(b,c), \text{path}_2(a,c)$

Model $M \models \Psi$

Derive input

- $\text{link}(a,b), \text{link}(b,c)$

Datalog input $I$
Generating Constraints

\[ link(N1, A, ?) \]
...

\[ path(Router, Net, Net, Cost) \leftarrow link(Router, Net, Cost) \]
\[ path(Router, Net, NextHop, C_1 + C_2) \leftarrow link(Router, NextHop, C_1), \]
\[ path(NextHop, Net, X, C_2) \]

\[ sp(Router, Net, NextHop, min(C)) \leftarrow path(Router, Net, NextHop, C) \]

\[ fwd(Router, Net, NextHop) \leftarrow sp(Router, Net, NextHop, C) \]

\[ fwd(A, N_2, D) \land fwd(A, N_3, B) \land ... \]

\[ \forall X, Y. \ path_1(X, Y) \iff \ link(X, Y) \]
\[ \forall X, Y. \ path_2(X, Y) \iff \left( \ link(X, Y) \lor \exists Z. \ (\ link(X, Z) \land \ path_1(Z, Y)) \right) \]
\[ path_2(a, c) \land \neg \ link(a, c) \]
Generating Constraints

\[ \text{link}(N1, A, ?) \]

\[ \ldots \]

\[ \text{path}(\text{Router}, \text{Net}, \text{Net}, \text{Cost}) \leftarrow \text{link}(\text{Router}, \text{Net}, \text{Cost}) \]

\[ \text{path}(\text{Router}, \text{Net}, \text{NextHop}, C_1 + C_2) \leftarrow \text{link}(\text{Router}, \text{NextHop}, C_1), \text{path}(\text{NextHop}, \text{Net}, X, C_2) \]

\[ \text{sp}(\text{Router}, \text{Net}, \text{NextHop}, \text{min}(C)) \leftarrow \text{path}(\text{Router}, \text{Net}, \text{NextHop}, C) \]

\[ \text{fwd}(\text{Router}, \text{Net}, \text{NextHop}) \leftarrow \text{sp}(\text{Router}, \text{Net}, \text{NextHop}, C) \]

\[ \text{fwd}(A, N_2, D) \wedge \text{fwd}(A, N_3, B) \wedge \ldots \]
Synthesis Algorithm (Naïve Approach)

Box 1

Box 2

Box 3

Box 4

Box 5

$M_{box4} \models \psi \wedge I_5$

$M_{box5} \models \psi$
Synthesis Algorithm (Naïve Approach)

Naïve approach does NOT work

1. Not all valid SMT models are valid network configurations.
2. The search space of valid inputs is huge.
Synthesis Algorithm (Our Approach)

**Input Constraints**

*Ensure the synthesized configurations are valid*

*Ex. OSPF Link costs are positive numbers*
Synthesis Algorithm (Our Approach)

Input Constraints
Ensure the synthesized configurations are valid
Ex. OSPF Link costs are positive numbers

Domain specific constraints
Reduce the search space
Ex. Routers only forward to connected neighbours
Synthesis Algorithm (Our Approach)

Input Constraints
Ensure the synthesized configurations are valid
Ex. OSPF Link costs are positive numbers

Domain specific constraints
Reduce the search space
Ex. Routers only forward to connected neighbours

Learn from mistakes
Don’t reuse the same invalid input

\[ M_{boxi} \models \psi \land l_{i+1} \land \neg \neg l_i \]
Synthesis Algorithm (Our Approach)

Input Constraints
Ensure the synthesized configurations are valid
Ex. OSPF Link costs are positive numbers

Domain specific constraints
Reduce the search space
Ex. Routers only forward to connected neighbours

Learn from mistakes
Don’t reuse the same invalid input
$M_{box_i} \models \psi \land l_{i+1} \land \neg l_i$

Partial evaluation
Don’t compute already known values
Network-wide Configuration Synthesis

Encode as a Datalog program

Datalog input identifies correct configuration

Routing requirements

Constraints on the forwarding plane computed by the routers

Automatically configure routers with synthesis

Via reduction to input synthesis for Datalog
Implementation
Implementation

The SyNET system ([http://synet.ethz.ch](http://synet.ethz.ch))

- Written in **Python** ($\approx 4K$ lines of code)
- Protocols encoded in **stratified Datalog** ($\approx 100$ rules)
- Uses the **Z3** constraint solver
- Outputs **Cisco** configurations
- Supports **iBGP**, singe-area **OSPF**, and **static routes**

Network-specific optimizations

- Partial evaluator for Datalog
- Protocol-specific constraints

Sample **Cisco** configuration output by SyNET

```
! A snippet from router A
interface f0/1
  ip address 10.0.0.2 255.255.255.254
  ip ospf cost 10
  description "To B"
interface f0/0
  ip address 10.0.0.0 255.255.255.254
  ip ospf cost 65530
  description "To C"
interface f1/0
  ip address 10.0.0.4 255.255.255.254
  ip ospf cost 65530
  description "To D"
```
Experiments
US-based network connecting major universities and research institutes

<table>
<thead>
<tr>
<th>Protocols / # Traffic classes</th>
<th>1 class</th>
<th>5 classes</th>
<th>10 classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Static + OSPF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Static + OSPF + BGP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Synthesis Times
**Experiment**

![Diagram of the US-based network connecting major universities and research institutes](image)

**Protocols / # Traffic classes**

<table>
<thead>
<tr>
<th>Protocols / Traffic classes</th>
<th>1 class</th>
<th>5 classes</th>
<th>10 classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td>1.3s</td>
<td>2.0s</td>
<td>4.0s</td>
</tr>
<tr>
<td>Static + OSPF</td>
<td>9.0s</td>
<td>21.3s</td>
<td>49.3s</td>
</tr>
<tr>
<td>Static + OSPF + BGP</td>
<td>13.3s</td>
<td>22.7s</td>
<td>1m 19.7s</td>
</tr>
</tbody>
</table>

**Synthesis Times**
Scalability Experiment

- Grid topologies with up to 64 routers
- Requirements for 10 traffic classes

Synthesis Time [s]

- < 1h for static routes
- < 24h for Static + OSPF
- < 24h for Static + OSPF + BGP

Grid topologies with up to 64 routers
Requirements for 10 traffic classes
SyNET: Programmable Networks with Synthesis

Global requirements vs local configurations

Network-wide configuration synthesis

Approach scales to realistic problems

http://synet.ethz.ch
<table>
<thead>
<tr>
<th>Routers/TC</th>
<th>Static</th>
<th>Static + OSPF</th>
<th>Static + OSPF + BGP</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>1.3 9.4 15.3</td>
<td>2 19.8 27.7</td>
<td>2.8 39.9 60.5</td>
</tr>
<tr>
<td>16</td>
<td>5.9 43.5 56</td>
<td>7.8 79.8 144.8</td>
<td>11.2 245.8 509</td>
</tr>
<tr>
<td>25</td>
<td>32 175.2 236.3</td>
<td>37 423.8 526.3</td>
<td>46.1 956.4 2409.3</td>
</tr>
<tr>
<td>36</td>
<td>169.7 600.5 854</td>
<td>181.5 1438.9 2618</td>
<td>207 4298.2 9311.7</td>
</tr>
<tr>
<td>49</td>
<td>740.2 1451.6 5000.7</td>
<td>782.3 5400.3 8118</td>
<td>850.7 19375.8 14746</td>
</tr>
<tr>
<td>64</td>
<td>2796.2 2773.2 6395</td>
<td>2843.8 20578.9 30291.3</td>
<td>2962.2 76396 281235</td>
</tr>
</tbody>
</table>
Analysis of Network Configurations in Datalog
Analysis of Network Configurations in Datalog

**Analysis question:**
Does the network $N$ configured with $C$ satisfy the requirements $R$?

**Query entailment:**
Does $P, I \models Q$ hold?

**Theorem:** Query entailment in Datalog is in PTIME

- **Network-wide configuration $C$** (protocol configurations for routers)
- **Network specification $N$** (OSPF, BGP, MPLS, ...)
- **Routing requirements $R$** (isolation, reachability, ...)

- **Datalog input $I$**
- **Datalog program $P$**
Datalog Stratification

**Input**

\[ \text{link}(N_1, a, 10) \]
...

**Program**

\[ \text{path}(\text{Router}, \text{Net}, \text{Net}, \text{Cost}) \leftarrow \text{link}(\text{Router}, \text{Net}, \text{Cost}) \]
\[ \text{path}(\text{Router}, \text{Net}, \text{NextHop}, C_1 + C_2) \leftarrow \text{link}(\text{Router}, \text{NextHop}, C_1), \text{path}(\text{NextHop}, \text{Net}, X, C_2) \]
\[ \text{sp}(\text{Router}, \text{Net}, \text{NextHop}, \text{min}(C)) \leftarrow \text{path}(\text{Router}, \text{Net}, \text{NextHop}, C) \]
\[ \text{fwd}(\text{Router}, \text{Net}, \text{NextHop}) \leftarrow \text{sp}(\text{Router}, \text{Net}, \text{NextHop}, C) \]

**Query**

\[ \text{fwd}(A, N_2, ?) \]